Pour qu'il y ait un intérêt pratique à participer à ce contrôle collectif sans lequel la cryptomonnaie n'existe plus, une récompense est donnée à intervalles réguliers à l'une des machines tenant les comptes. Aujourd'hui par exemple, 25 *bitcoins* sont distribués toutes les dix minutes par le protocole *Bitcoin*. Cette machine gagnante est désignée par un procédé fondé sur une preuve de travail dont le fonctionnement doit être parfait pour que le protocole résiste aux attaques. C'est la plus belle application à ce jour de l'idée des preuves de travail, et de nombreuses caractéristiques de *Bitcoin* en dépendent. Donnons quelques précisions.

Chaque machine candidate essaie de résoudre un problème dont les paramètres sont fixés par le protocole *Bitcoin* et qui est formulé sans intervention d'un service centralisé. Il s'agit d'un problème du type « trouver un $x$ tel que $f_p(x) < y$ » évoqué plus haut. La fonction $f$ à sens unique est connue de tous, et c'est la fonction de hachage SHA-256. On l'utilise beaucoup en cryptographie et les spécialistes s'accordent à considérer qu'elle a les propriétés voulues pour servir dans une preuve de travail. La première machine qui résout le problème gagne la compétition et emporte les 25 *bitcoins*. Le protocole ajuste régulièrement le seuil donné par le nombre $y$, qui fixe la difficulté de la preuve de travail en prenant en compte les concours précédents, de telle façon que le temps moyen entre deux concours soit de dix minutes environ.

## Des preuves de travail pour les bitcoins

Cette émission contrôlée, prévisible et faible de *bitcoins* toutes les dix minutes est la seule façon d'en créer. Cela a pour intérêt que les détenteurs de *bitcoins* sont certains de ne pas être victimes des phénomènes inflationnistes provoqués par l'émission massive de devises, comme c'est parfois le cas pour les monnaies contrôlées par les banques centrales.

Participer à ce concours porte le nom de « minage », car le travail de calcul fait par les machines évoque celui d'un mineur dans une mine d'or, qui le conduit à s'enrichir s'il a la chance de trouver une pépite. Les informaticiens propriétaires des machines participant à ce concours permanent, répété toutes les dix minutes, se nomment entre eux des « mineurs ».

Il est important de comprendre que puisqu'il n'y a pas d'autorité centrale gérant les *bitcoins* (il en va de même pour les autres cryptomonnaies), l'acceptation de la preuve de travail se fait collectivement par tous les mineurs. Quand l'un d'eux a réussi à relever le défi de trouver un $x$ tel que $f_p(x) < y$, il envoie un message sur le réseau, qui prévient les autres mineurs. Ceux-ci vérifient alors, chacun de son côté, que la solution proposée est correcte et se remettent au travail pour résoudre le défi suivant.

Le temps nécessaire pour qu'un gagnant soit trouvé n'est pas fixé avec précision : la

## 2. Le hachage

Une fonction de hachage associe, à un $x$ donné (une suite de caractères ou un nombre) de longueur quelconque une valeur $f(x)$ de taille fixe nommée « empreinte de $x$ » ou « condensé de $x$ » ou *hash*. Le passage de $x$ à $f(x)$ se fait par une série d'opérations qui commencent en général en découpant $x$ en petits morceaux (d'où le nom de hachage), qui sont ensuite rassemblés et emmêlés de façon à obtenir un mélange aussi parfait que possible. Pour la fonction classique de hachage SHA-256, la longueur de $f(x)$ est de 256 bits ; il y a donc $2^{256} = 1{,}157 \times 10^{77}$ valeurs possibles, ce qui est considérable.

Les fonctions de hachage à usage cryptographique ont les propriétés suivantes.

*(a)* Elles sont « à sens unique » : le passage de $x$ à $f(x)$ se calcule facilement et rapidement, mais pour un $y$ donné, trouver un $x$ tel que $y = f(x)$ est impossible en pratique.

*(b)* Il est impossible en pratique de trouver deux valeurs $x$ et $x'$ différentes telles que $f(x) = f(x')$.

*(c)* Elles produisent des valeurs assimilables à des valeurs aléatoires uniformes (prises dans l'ensemble des valeurs possibles) et, en particulier, même si $x$ et $x'$ sont deux suites de caractères n'ayant qu'un seul caractère différent, les valeurs $f(x)$ et $f(x')$ n'ont aucun lien apparent.

Voici un exemple. La donnée $x$ est transformée en une suite de 0 et de 1 qui est découpée en morceaux de longueur 8. Les morceaux sont ensuite additionnés comme des nombres en base 2. On ne garde du résultat que les quatre chiffres en positions 3, 4, 5 et 6 à partir de la droite :
$x$=01010010100101010010010101 0
→01010010+10010101+00100101 + 010 = 100001110 → $f(x)$ = 0011.

Cet exemple est bien sûr trop simple pour avoir les propriétés *(a)*, *(b)* et *(c)*. La fonction de hachage SHA-256 mélange soigneusement les bits de la donnée $x$ selon un algorithme public (voir http://en.wikipedia.org/wiki/SHA-2).

Les fonctions de hachage cryptographiques ont de multiples usages dont voici trois exemples.

– Elles permettent d'assurer l'intégrité d'un gros fichier $x$, auquel on lui associera son empreinte $f(x)$, beaucoup plus courte. Celui qui téléchargera $x$ s'assurera que $x$ lui parvient sans erreur et en totalité en calculant l'empreinte du fichier qu'il aura reçu, qui doit être la même que celle publiée sur le site. Si $x$ a mal été copié, il le saura, car la probabilité que la modification n'ait pas changé l'empreinte est infinitésimale.

– Au lieu de mémoriser les mots de passe d'accès aux comptes sur un service en ligne, celui-ci ne mémorise que les empreintes des mots de passe. Le serveur peut vérifier que les utilisateurs donnent les bons mots de passe, mais si un pirate accède à sa liste d'empreintes, il ne pourra rien en faire.

– Pour produire une suite de nombres aléatoires dépendant d'un fichier $x$, on calculera par exemple $f(x.1)$, $f(x.2)$,... où $x.i$ désigne le fichier $x$ suivi de l'écriture du nombre $i$ (par exemple en binaire). En changeant le $x$, on a autant de suites de nombres aléatoires qu'on veut (ce qui permet par exemple de chiffrer des messages).

L'importance des fonctions de hachage est telle, en cryptographie, qu'un grand soin est mis à les élaborer et à les soumettre à toutes sortes de tests et d'attaques. Le NIST, aux États-Unis, propose des fonctions bien contrôlées. La fonction SHA-256 utilisée par *Bitcoin* en est une. Le NIST publie aussi des recommandations concernant le choix et l'usage des fonctions de hachage. Il a récemment organisé une compétition pour concevoir une nouvelle génération de fonctions de hachage qui a abouti à la norme SHA-3 en 2012. Voir : http://www.nist.gov/itl/csd/ct/hash_competition.cfm.

## 3. L'inversion d'une fonction de hachage

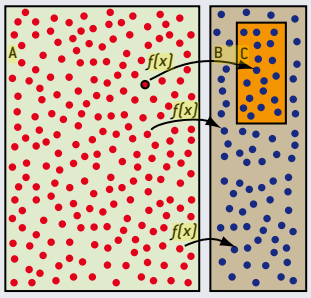
Explicitons une méthode pour définir et ajuster l'énigme d'une preuve de travail.

La fonction $f$ de hachage transforme une suite $x$ de 0 et de 1 de longueur quelconque en une autre, $y$, de longueur 256 (comme le SHA-256). Puisqu'il est impossible en pratique de trouver pour un $y$ donné un $x$ tel que $f(x) = y$ (inversion complète de $f$), on crée une énigme plus raisonnable et exigeant un travail faisable en ne demandant qu'une inversion partielle de $f$ : trouver un $x$ tel que $f(x)$ soit une suite de caractères commençant par $k$ fois '0' (donc une suite de la forme 0 0 0 ... 0 $a_1$ $a_2$ $a_3$... $a_n$).

Plus $k$ est grand, plus il sera difficile de trouver un $x$ satisfaisant. On peut même être plus précis : pour trouver un tel $x$, il faut en moyenne essayer environ $2^k$ valeurs de $x$. L'énigme « trouver un $x$ tel que $f(x)$ commence par dix fois '0' » exige donc environ $2^{10} = 1024$ calculs de $f(x)$. Pour 20 fois '0', il en faudrait un million environ. En choisissant $k$, on ajuste donc la difficulté de l'énigme qu'on formule. On a là une méthode rapide pour formuler des énigmes de « preuve de travail » dont la difficulté est facile à évaluer.

Dans les preuves de travail (pour se protéger des spams, ou interdire les attaques par déni de service), il faut éviter que les mêmes énigmes soient soumises plusieurs fois. Si c'était le cas, des bibliothèques de solutions seraient calculées par avance et les attaquants, au lieu de faire les calculs attendus, se contenteraient d'aller y puiser les solutions des problèmes qu'on leur soumet.

Pour éviter cela, on fait dépendre l'énigme posée d'un paramètre $p$. On demandera par exemple de trouver un $x$ commençant par la chaîne $p = 0100011010100$ et tel que $f(x) < y$.




*La fonction f envoie les éléments x de l'ensemble A (grand) dans l'ensemble B (petit). Inverser partiellement f, c'est trouver un élément x tel que f(x) soit dans C. Plus C est petit, plus la tâche est difficile.*

Le nombre de paramètres possibles $p$ étant illimité, l'idée de constituer des bibliothèques de solutions calculées par avance n'est plus applicable.

Les énigmes possibles posées par la méthode de l'inversion partielle des fonctions de hachage sont donc à la fois parfaitement ajustables, et si nombreuses que celui à qui on les propose ne peut que se soumettre et faire le travail de les résoudre en essayant de très nombreux $x$, ce qui en moyenne lui prendra le temps qu'on aura choisi. Quand il aura trouvé la solution, il aura vraiment prouvé qu'il a travaillé !

Notons encore que si l'on est gêné par le fait que le temps de travail varie trop, selon que celui qui tente de résoudre le problème posé a de la chance ou pas (variance importante), on peut remplacer la demande de résoudre un problème, par la demande de résoudre plusieurs petits problèmes, ce qui a pour effet de diminuer la variance du temps nécessaire pour aboutir.

durée moyenne est de dix minutes, mais elle sera plus longue ou plus courte selon que les mineurs auront eu de la chance ou pas dans leur tentative de résolution du problème de la preuve de travail.

## Une course folle à la puissance...

Si chaque machine n'utilisait que sa propre puissance pour gagner, la probabilité pour chacune de gagner serait proportionnelle à sa puissance. Cependant, comme la fonction $f$ est fixée et connue de tous, le problème de trouver un $x$ tel que $f_p(x) < y$ est très particulier. Les personnes intéressées par le minage des *bitcoins* se sont mises à utiliser des puces conçues spécialement pour résoudre rapidement le type particulier de problèmes posés dans les preuves de travail de *Bitcoin*. Une course folle à la puissance de minage en a résulté, conduisant à une véritable industrie du minage !

La puissance de minage d'un système informatique se mesure en comptant le nombre de calculs de type $x \to f(x)$ que le système est capable de faire par seconde, pour la fonction SHA-256. On est ainsi passé de $24 \times 10^{12}$ calculs de valeurs de $f$ par seconde pour l'ensemble des mineurs en janvier 2013 à $11 \times 10^{15}$ en janvier 2014, soit une multiplication par plus de 500 en un an.

Le coût en électricité de ces minages (coût des « preuves de travail ») a été évalué. Certains ont calculé qu'il est de plus de dix millions d'euros par jour. C'est plus que la valeur des *bitcoins* gagnés chaque jour (environ deux millions d'euros). Les spécialistes ne sont pas d'accord sur ce nombre de dix millions, mais, même imprécis, il signifie qu'être mineur de *bitcoins* aujourd'hui n'est pas nécessairement rentable. Acheter du matériel et dépenser de l'électricité en le faisant fonctionner est un pari risqué, dont le succès dépend de deux facteurs très difficiles à contrôler ou même à anticiper : la puissance de calcul de vos concurrents, et le cours extrêmement volatil du *bitcoin*.

Notons que certains pirates informatiques ont conçu des programmes, nommés chevaux de Troie, qui s'installent par le biais du réseau sur une multitude d'ordinateurs (le vôtre peut-être) et leur font exécuter le minage. Bien évidemment, lorsque le minage réussit, la solution est transmise au pirate qui, pour son propre compte, touche les 25 *bitcoins*. C'est vous qui payez l'électricité, mais c'est lui qui empoche le bénéfice !

## ...et un énorme gâchis ?

Parmi les nombreuses critiques formulées à l'encontre de *Bitcoin*, l'une d'elles concerne justement les preuves de travail qui définissent le minage. Les sommes d'argent considérables dépensées en électricité et pour fabriquer du matériel spécialisé calculant le plus rapidement possible des $f_p(x)$ semblent absurdes, car en soi de tels calculs ne servent à rien. N'est-ce pas là un gâchis terrifiant, engendrant par ailleurs une importante pollution liée à l'électricité consommée, qu'il faut bien produire ?

Une double réponse a été donnée à cette critique. D'une part, il a été répliqué

que les moyens dépensés pour le minage n'étaient pas vains, puisqu'ils permettent à ces monnaies de fonctionner. Fabriquer des billets de banques, les transporter dans des véhicules blindés, les enfermer dans des coffres que l'on fait garder : tout cela coûte cher et pourtant personne ne conteste la nécessité de ces dépenses puisqu'elles servent à faire fonctionner les monnaies fiduciaires habituelles émises par les banques centrales. Le *bitcoin* n'a pas besoin d'être imprimé (c'est une monnaie numérique), il n'y a pas besoin de voitures blindées pour son transport, mais il faut que sa gestion soit assurée. Le minage assure en partie cette gestion, qui n'est donc que l'équivalent des dépenses de fonctionnement des autres monnaies et n'a finalement rien d'absurde.

## Concevoir des preuves de travail utiles

L'autre réponse est plus intéressante, elle est aujourd'hui encore en discussion. Les preuves de travail les plus habituelles (dont celle utilisée par *Bitcoin*) consistent à faire des calculs pour résoudre un problème qui, malheureusement, est sans intérêt réel : une fois que l'on a trouvé un $x$ tel que $f_p(x) < y$, ce $x$ n'est utile à personne et ne le sera sans doute jamais. Ne pourrait-on pas concevoir des preuves de travail fondées sur des problèmes dont la solution serait utile ?

On le sait, les mathématiciens aiment les nombres premiers dont la connaissance progresse et est utile (en particulier en cryptographie !). Divers groupes de nombres premiers sont jugés intéressants. Les paires de nombres premiers jumeaux sont les paires $(p, p + 2)$ de nombres premiers ayant un écart de deux unités, par exemple (17, 19). On cherche toujours à prouver qu'il en existe une infinité. Il y a aussi les paires de nombres premiers de Sophie Germain (mathématicienne française qui vécut de 1776 à 1831) qui sont de la forme $(p, 2p + 1)$, par exemple (3, 7), (5, 11), (11, 23). Elles conduisent aux chaînes de Cunningham $(p_1, p_2, ..., p_k)$ où chaque couple $(p_i, p_{i+1})$ est une paire de nombres de Sophie Germain : $p_2 = 2p_1 + 1$, $p_3 = 2p_2 + 1$, ..., $p_{k+1} = 2p_k + 1$, chaque $p_i$ étant un nombre premier.

Des chercheurs tiennent à jour des sites Internet qui indiquent les records de longueur des chaînes de Cunningham. Sunny King a montré en 2013 qu'une telle recherche peut servir de base à des preuves de travail aux propriétés à peu près équivalentes à celles provenant des fonctions à sens unique.

S. King a créé en juillet 2013 sa propre cryptomonnaie, *Primecoin*, proche dans sa conception de *Bitcoin*, mais utilisant des preuves de travail qui conduisent à des chaînes de Cunningham nouvelles. Des chaînes de Cunningham records, de longueur 10, 11, 12 et 13, ont ainsi été découvertes, ce qui prouve l'efficacité de la méthode proposée et sa capacité à contribuer à la recherche mathématique.

Précisons toutefois que l'intérêt de connaître des chaînes de Cunningham n'est pas très grand, et que la sûreté des preuves de travail fondées sur leur recherche n'est pas aussi bonne que celle des preuves de travail fondées sur les fonctions à sens unique. Les utilisateurs des cryptomonnaies ont cependant fait bon accueil à cette nouveauté. Parmi toutes les cryptomonnaies existantes (il y en a aujourd'hui plus de 80, voir http://coinmarketcap.com/), *Primecoin* se classe onzième pour la capitalisation. Le total des *primecoins* émis vaut dix millions de dollars (le 7 février 2014). Qui a dit que les mathématiques n'étaient pas un bon moyen de gagner de l'argent ?

Un autre projet en cours de développement, nommé *Curecoin,* tente de concevoir une preuve de travail (et une cryptomonnaie associée) qui soit beaucoup plus clairement utile. Le calcul mené par les machines de-

### 4. Des preuves de travail utiles

Les preuves de travail utilisées aujourd'hui ont un côté absurde : on demande aux ordinateurs de résoudre des problèmes mathématiques qui n'ont aucun intérêt. Sunny King (pseudonyme d'une personne ou d'un groupe) a conçu une preuve de travail aussi facile à ajuster que l'inversion partielle des fonctions de hachage cryptographiques et qui permet de trouver des chaînes de nombres premiers intéressantes : les chaînes de Cunningham.

Ce sont des suites de nombres premiers $(p_1, p_2, ..., p_k)$ vérifiant : $p_2 = 2p_1 + 1$, $p_3 = 2p_2 + 1$, ..., $p_{k+1} = 2p_k + 1$. Ces chaînes de Cunningham sont reliées aux nombres premiers de Sophie Germain, une mathématicienne française (1776-1831) : un nombre premier $p$ est un nombre premier de Sophie Germain si $2p + 1$ est aussi un nombre premier.

Ces nombres premiers particuliers ont un intérêt en raison d'un théorème de Sophie Germain lié au théorème de Fermat. Dans les chaînes de Cunningham, tous les nombres sont des nombres de Sophie Germain, sauf le dernier.

L'utilisation de ces preuves de travail où l'on recherche des nombres premiers est au cœur du fonctionnement de la cryptomonnaie *Primecoin*, concurrente de *Bitcoin*. Le travail de minage de *Primecoin* a aujourd'hui produit plusieurs chaînes nouvelles intéressantes. La chaîne de Cunningham de longueur 11 comportant les plus grands nombres premiers est l'une d'elles. Le premier nombre de cette chaîne record découverte le 3 août 2013 comporte plus de 200 chiffres. Il s'écrit :

$p_1 = k \times 151\# - 1$,

où $k$ = 73 853 903 764 168 979 088 206 401 473 739 410 396 455 001 112 581 722 569 026 969 860 983 656 346 568 919

et où 151# désigne le produit des nombres premiers inférieurs ou égaux à 151 (voir http://users.cybercity.dk/~dsl522332/math/Cunningham_Chain_records.htm).

Sophie Germain à l'âge de 14 ans, portrait de Auguste Eugène Leray.